WHAT IS CLAIMED IS

1. A method of analyzing intersections between objects in computer animation comprising the steps of:

providing a plurality of objects represented by a plurality of meshes, with each of said plurality of objects being represented by one of said plurality of meshes and each of said meshes being formed by a set of vertices, where a set of pairs of vertices of said set of vertices define a set of edges;

checking all edges of said meshes to determine if said set of edges of said meshes intersect with any of said plurality of meshes;

tracing an intersection path formed by intersection of said edges with any of said plurality of meshes; and

determining which vertices of said meshes are contained within said intersection path and setting a polarity of each of said contained vertices to indicate that those vertices are contained within said intersection path.

2. A method of analyzing intersections between objects according to claim 1 wherein said step of determining which vertices of said meshes are contained within said intersection path comprises examining vertices of a mesh that contains said intersection path within a certain distance from a particular edge of said intersection path and characterizing said vertices to determine which vertices of said meshes are contained within said intersection path.

3. A method of analyzing intersections between objects according to claim 1 wherein said step of determining which vertices of said meshes are contained within said intersection path comprises the steps of:

selecting an arbitrary edge of a mesh that crosses said intersection path, where said arbitrary edge is formed by vertices *u* and *v* and where said mesh contains said intersection path;

performing a search of said mesh, radiating from one of said vertices *u* and *v*, identifying all vertices in all of said edges that cross said intersection path, and defining the set of vertices identified as a playpen;

tracing said intersection path and identifying vertices, within said playpen, on a left side of said intersection path as left and retracing said intersection path in an opposite direction and identifying vertices, with said playpen, on a right side of said intersection path as right;

determining whether vertices adjacent to vertices identified as left and right lie outside of said playpen;

discarding said intersection path when at least one of both left and right identified vertices lie outside said playpen;

determining whether at least one vertex adjacent to said right identified vertices lies outside said playpen; and

changing said polarity of each of said left identified vertices to indicate that those vertices are contained within said intersection path when at least one vertex adjacent to said right identified vertices lies outside said playpen and changing said polarity of each of said right identified vertices to indicate that those vertices are contained within said intersection path when at least one vertex adjacent to said left identified vertices lies outside said playpen.

4. A method of analyzing intersections between objects according to claim 1 wherein said intersection path is a self-intersection with the intersection path being

contained in a single mesh and said step of setting a polarity of each of said contained vertices to indicate that those vertices are contained within said intersection path comprises setting the color of said vertices that are contained within said intersection path to a predetermined color when the intersection yields one region and setting the color of vertices of a first portion of said single mesh contained within said intersection path to a first color and setting the color of vertices of a second portion of said single mesh contained within said intersection path to a second color when the intersection yields two unconnected regions.

5. A method of analyzing intersections between objects according to claim 1 wherein said intersection path is an intersection between a first mesh and a second mesh and said step of setting a polarity of each of said contained vertices to indicate that those vertices are contained within said intersection path comprises setting the color of vertices of the first mesh contained within said intersection path to a first color and setting the color of vertices of the second mesh contained within said intersection path to a second color.

6. A method of analyzing intersections between objects according to one of claims 4 and 5, further comprising displaying said objects on a computer display with vertices colored as said vertices have been set.

7. A method of determining pinching between objects in computer animation comprising the steps of:

providing a plurality of objects represented by a plurality of meshes, with each of said plurality of objects being represented by one of said plurality of meshes and

each of said meshes being formed by a set of vertices;

analyzing intersections between said objects and changing a polarity of each of a plurality vertices contained in an intersection path created by an intersection of said plurality of meshes;

selecting a particular vertex of said set of vertices bound between surfaces of said objects and closer to one of said surfaces, where said surfaces have defined insides and outsides and said particular vertex is inside of both surfaces;

determining whether any vertices inside of said surfaces have their polarities set; and

indicating that said particular vertex is pinched when any vertices inside of said surfaces have their polarities set.

8. A method of determining pinching between objects in computer animation according to claim 7 further comprising the step of:

constraining motion of said pinched particular vertex when motion in said computer animation is simulated.

9. A method of determining pinching between objects in computer animation according to claim 7 wherein said step of changing a polarity of each of a plurality vertices contained in an intersection path comprises setting the color of each of a plurality vertices contained in an intersection path and further comprising the step of:

displaying said objects on a computer display with vertices colored as said vertices have been set such that an animator can see the intersection and pinching of said objects.

10. A method of simulating motion of objects in computer animation comprising the steps of:

providing a plurality of objects represented by a plurality of meshes, with each of said plurality of objects being represented by one of said plurality of meshes and each of said meshes being formed by a set of vertices, where at least one of said objects is an animated object and at least one of said objects is a simulated object;

positioning said objects at some time t to provide one frame of said computer animation;

analyzing intersections between said objects and setting a polarity of each of a plurality vertices contained in an intersection path created by an intersection of said plurality of meshes;

setting a simulated force between vertices of said at least one simulated object based on the polarity set for said vertices of said at least one simulated object; and

advancing the computer animation to a time $t + \Delta t$ and simulating motions of said objects using said simulated force to simulate motions of said at least one simulated object.

11. A method of simulating motion of objects in computer animation according to claim 10 wherein

when said intersection path is a self-intersection with the intersection path contained in a single mesh, said step of setting said polarity of each of said plurality vertices contained in said intersection path comprises setting the color of each of said plurality vertices to a predetermined color when the intersection yields one region and setting the color of vertices of a first portion of said single mesh contained within said intersection path to a first color and setting the color of vertices of a second portion of

said single mesh contained within said intersection path to a second color when the intersection yields two unconnected regions, and

when said intersection path is an intersection between a first mesh and a second mesh and said step of setting a polarity of each of said plurality of vertices contained in said intersection path comprises setting the color of each of said plurality vertices of the first mesh to said first color and setting the color of each of said plurality vertices of the second mesh to said second color.

12. A method of simulating motion of objects in computer animation according to claim 11 wherein said step of setting a simulated force between vertices of said at least one simulated object comprises:

setting said simulated force to cause an attraction between vertices of said at least one simulated object when said vertices are set to said first or second colors;

setting said simulated force to cause an repulsion between vertices of said at least one simulated object when said vertices are not set to said first, second, or predetermined colors;

setting said simulated force to cause neither attraction or repulsion between vertices of said at least one simulated object when said vertices are set to said predetermined color.

13. A computer program product comprising:

a computer usable medium having computer readable program code means embodied in said medium for causing a computer to manipulate and analyze computer generated objects, said computer readable program code means comprising:

means for providing a plurality of objects represented by a plurality of meshes, with each of said plurality of objects being represented by one of said plurality of meshes and each of said meshes being formed by a set of vertices, where a set of pairs of vertices of said set of vertices define a set of edges;

means for checking all edges of said meshes to determine if said set of edges of said meshes intersect with any of said plurality of meshes;

means for tracing an intersection path formed by intersection of said edges with any of said plurality of meshes; and

means for determining which vertices of said meshes are contained within said intersection path and setting a polarity of each of said contained vertices to indicate that those vertices are contained within said intersection path.

14. A computer program product according to claim 13 wherein said means for determining which vertices of said meshes are contained within said intersection path comprises means for examining vertices of a mesh that contains said intersection path within a certain distance from a particular edge of said intersection path and means for characterizing said vertices to determine which vertices of said meshes are contained within said intersection path.

15. A computer program product according to claim 13 wherein said means for determining which vertices of said meshes are contained within said intersection path comprises:

means for selecting an arbitrary edge of a mesh that crosses said intersection path, where said arbitrary edge is formed by vertices u and v and where said mesh contains said intersection path;

means for performing a search of said mesh, radiating from one of said vertices *u* and *v*, identifying all vertices in all of said edges that cross said intersection path, and defining the set of vertices identified as a playpen;

means for tracing said intersection path and identifying vertices, within said playpen, on a left side of said intersection path as left and retracing said intersection path in an opposite direction and identifying vertices, with said playpen, on a right side of said intersection path as right;

means for determining whether vertices adjacent to vertices identified as left and right lie outside of said playpen;

means for discarding said intersection path when at least one of both left and right identified vertices lie outside said playpen;

means for determining whether at least one vertex adjacent to said right identified vertices lies outside said playpen; and

means for changing said polarity of each of said left identified vertices to indicate that those vertices are contained within said intersection path when at least one vertex adjacent to said right identified vertices lies outside said playpen and changing said polarity of each of said right identified vertices to indicate that those vertices are contained within said intersection path when at least one vertex adjacent to said left identified vertices lies outside said playpen.

16. A computer program product according to claim 13 wherein said intersection path is a self-intersection with the intersection path being contained in a single mesh and said means for setting a polarity of each of said contained vertices to indicate that those vertices are contained within said intersection path comprises means for setting the color of said vertices that are contained within said intersection

path to a predetermined color when the intersection yields one region and means for setting the color of vertices of a first portion of said single mesh contained within said intersection path to a first color and setting the color of vertices of a second portion of said single mesh contained within said intersection path to a second color when the intersection yields two unconnected regions.

17. A computer program product according to claim 13 wherein said intersection path is an intersection between a first mesh and a second mesh and said means of setting a polarity of each of said contained vertices to indicate that those vertices are contained within said intersection path comprises means for setting the color of vertices of the first mesh contained within said intersection path to a first color and setting the color of vertices of the second mesh contained within said intersection path to a second color.

18. A computer program product according to one of claims 16 and 17, further comprising means for displaying said objects on a computer display with vertices colored as said vertices have been set.

19. A computer program product comprising:

a computer usable medium having computer readable program code means embodied in said medium for causing a computer to manipulate and analyze computer generated objects in computer animation, said computer readable program code means comprising:

means for providing a plurality of objects represented by a plurality of meshes, with each of said plurality of objects being represented by one of said plurality of

meshes and each of said meshes being formed by a set of vertices;

means for analyzing intersections between said objects and means for changing a polarity of each of a plurality vertices contained in an intersection path created by an intersection of said plurality of meshes;

means for selecting a particular vertex of said set of vertices bound between surfaces of said objects and closer to one of said surfaces, where said surfaces have defined insides and outsides and said particular vertex is inside of both surfaces;

means for determining whether any vertices inside of said surfaces have their polarities set; and

means for indicating that said particular vertex is pinched when any vertices inside of said surfaces have their polarities set.

20. A computer program product according to claim 19 further comprising:

means for constraining motion of said pinched particular vertex when motion in said computer animation is simulated.

21. A computer program product according to claim 19 wherein said means for changing a polarity of each of a plurality vertices contained in an intersection path comprises means for setting the color of each of a plurality vertices contained in an intersection path and further comprising:

means for displaying said objects on a computer display with vertices colored as said vertices have been set such that an animator can see the intersection and pinching of said objects.

22. A computer program product comprising:

a computer usable medium having computer readable program code means embodied in said medium for causing a computer to manipulate and analyze computer generated objects in computer animation, said computer readable program code means comprising:

means for providing a plurality of objects represented by a plurality of meshes, with each of said plurality of objects being represented by one of said plurality of meshes and each of said meshes being formed by a set of vertices, where at least one of said objects is an animated object and at least one of said objects is a simulated object;

means for positioning said objects at some time t to provide one frame of said computer animation;

means for analyzing intersections between said objects and setting a polarity of each of a plurality vertices contained in an intersection path created by an intersection of said plurality of meshes;

means for setting a simulated force between vertices of said at least one simulated object based on the polarity set for said vertices of said at least one simulated object; and

means for advancing the computer animation to a time $t + \Delta t$ and simulating motions of said objects using said simulated force to simulate motions of said at least one simulated object.

23. A computer program product according to claim 22 wherein

when said intersection path is a self-intersection with the intersection path contained in a single mesh, said means for setting said polarity of each of said plurality vertices contained in said intersection path comprises means for setting the color of

each of said plurality vertices to a predetermined color when the intersection yields one region and setting the color of vertices of a first portion of said single mesh contained within said intersection path to a first color and setting the color of vertices of a second portion of said single mesh contained within said intersection path to a second color when the intersection yields two unconnected regions, and

when said intersection path is an intersection between a first mesh and a second mesh said means for setting a polarity of each of said plurality of vertices contained in said intersection path comprises means for setting the color of each of said plurality vertices of the first mesh to said first color and setting the color of each of said plurality vertices of the second mesh to said second color.

24. A computer program product according to claim 23 wherein said means for setting a simulated force between vertices of said at least one simulated object comprises:

means for setting said simulated force to cause an attraction between vertices of said at least one simulated object when said vertices are set to said first or second colors;

means for setting said simulated force to cause an repulsion between vertices of said at least one simulated object when said vertices are not set to said first, second, or predetermined colors;

means for setting said simulated force to cause neither attraction or repulsion between vertices of said at least one simulated object when said vertices are set to said predetermined color.